International Journal of Engineering & Technology, 7 (4.1) (2018) 1-3

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Ranking and Closest Element Algorithms on Centralized Diamond Architecture

Masumeh Damrudi*, Kamal Jadidy Aval

Computer Science Department, Islamic Azad University, Firoozkooh Branch, 3981838381, Iran
**Corresponding author E-mail:m.damrudi@gmail.com*

Abstract

Employing an appropriate algorithm, hardware and technique makes operations easier and faster in today digital world. Searching data which is a fundamental operation in computer science is an important problem in different Areas. An efficient algorithm is useful to search a data element in huge amount of data while information is growing in every second. There are various papers on searching algorithms to find data elements whereas different types of query in different areas of works including position, rank, count and closest element exists. Each of these queries may be useful in different computations. This paper proposed two algorithms of these four types of query employing Centralized Diamond architecture which consume constant execution time.

Keywords: *closest element, diamond architecture, parallel, ranking, search.*

1. Introduction

Parallel systems have widely developed in recent years whereas information is growing and increasing massively. There are different algorithms and mechanism to perform an operation in parallel to decrease the execution time of an operation. Searching a data is an inevitable task in computational functions which needs to carry out instantaneously in different areas. The solution can be parallelizing a serial task or proposing a parallel algorithm or even an architecture to achieve a better performance. There are some search algorithms explained in [1]. A parallel searching algorithm on n×m matrices which is sorted has *O(loglogn)* time complexity and consumes *O(n/loglog n)* processors on a CREW PRAM [2]. A search algorithm employing tree architecture with order of $O(\log_2(n+1))$ is proposed in [3]. Two searching algorithms with different complexities are presented in parallel while apply different ways [4]. We issued a Parallel Search on Hypercube Interconnection Network with *O(log N)* time complexity [5], a searching algorithm (CS) [6] and the improvement of this algorithm [7] on Centralized Diamond Architecture. Selim G. Akl explained position, rank, count and closest element query algorithms on tree [8]. We described the counting and positioning algorithm in [7]. This paper presents other different types of querying including ranking and closest element on Centralized Diamond architecture with constant execution time.

2. Searching on Centralized Damon

The Centralized Diamond Architecture is considered to have *n=16* and *N=29* while *n* is the number of data as input and *N* is the number of processing elements that is called PE. This architecture is shown in Figure 1. The details about this architecture are described in our previous works such as [6, 7, 9, 10].

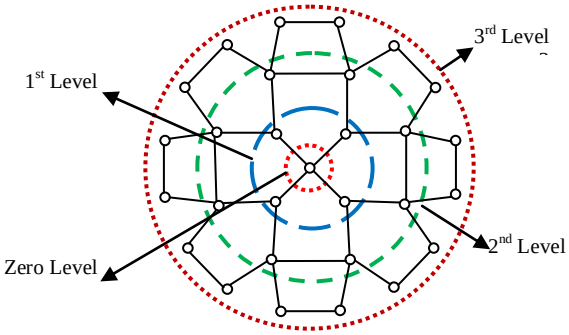

Fig. 1: Centralized Diamond architecture

Querying is an operation which a data is given as input and it is required to determine whether the data is in a given list. This is the basic search algorithm. Two other variations of this basic algorithm including ranking and closest element are discussed as follows.

2.1. Ranking of Searching Value

The aim of rank algorithm is to find that if the key value *k* exists in the input list. The rank of a data element is obtained by finding the number of data elements smaller than the *k*.
In this procedure, data elements as inputs are in the third level. The aim is to search the key value of the *k* and its location among data elements. The data elements are considered to be distinct.
First of all, *k* is sending to the processing elements in the third layer at once. Each PE produce a number which is one or zero. If their data is smaller than *k*, the PE produce one otherwise produce zero. All nodes send their results to the second layer. At the second level, PEs receive zeros and ones. They add their received values and send the results of their summations to their parent in the first level. The processing elements in this level carry out the same operation the PEs in the second level have done. They add

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

their inputs. The centralized result of addition will be sent to its parent which is the central node in the zero level. Zero level receives the outputs of first level. The centralized PE add the sum of the received values with one. The result of this operation produce the rank of the *k*.
It can be considered that the architecture is made up of four parts which work in parallel while the algorithm runs. It is shown in Figure 2.

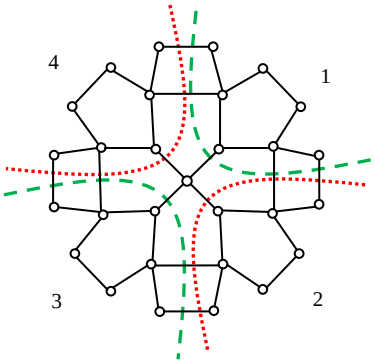

Fig. 2: Parts of Centralized Diamond

The procedure of ranking algorithm for one of these four parts is illustrated in Figure 3. In this figure the second part is shown.

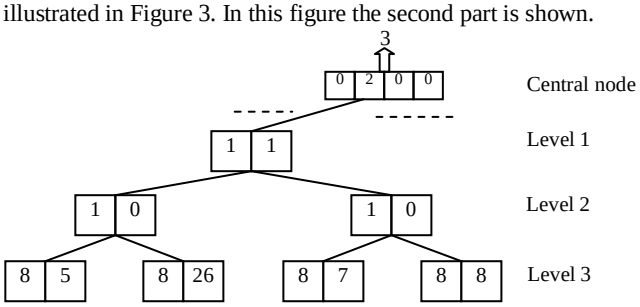

Fig. 3: The procedure of ranking algorithm

In this example the key value *k* is 8. It means that the result of the procedure is the rank of 8. PEs containing 5 and 7 are smaller than 8 in Figure 3, which produce one. In the second layer, the received data are added and the results will be send to the first layer. The PEs in the first layer also add their inputs and send the results to the centralized processing element. The zero level sum the addition of its inputs with one to produce the final result which is three for the values in Figure 3.

2.2. Closest Element in Searching

It is useful to find the closest element to the key value *k* in some applications. In this algorithm, first *k* is entered to the processing elements in the third layer of the architecture simultaneously. All nodes compute $x_i - k$ while they get value *k* and its data as x_i. The PEs in third layer produce pairs of $(i, x_i - k)$ and send them to their parents. Each processor receives two pairs of values $(i, x_i - k)$ and $(j, x_j - k)$ in the second level. The pair with the smaller value result will be sent to the upper level which is first level. All processing element receives two pairs of values $(i, x_i - k)$ and $(j, x_j - k)$, in the first level. Like the second level, the pair with smaller value will be sent to the intermediate node. The central node also sends the pair with smaller value as the final result. This shows the closest element to the key value *k*. It must be considered that to identify the smaller value, the absolute values of the subtraction operations are compared.
The procedure of closest element algorithm for second part of these four parts is illustrated in Figure 4.

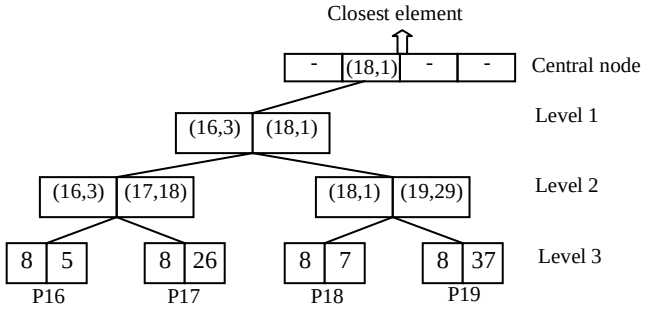

Fig. 4: The procedure of closest element algorithm

In this example the key value *k* is 8 as the previous example. It means that the result of the procedure is the closest value to 8. PEs from 16 to 19 for the second part of the architecture are shown in Figure 4. Each PE calculate the distance of its value with the received value and send upward the pair of the PE number with the calculated value to its parent. In the second layer, nodes select the smaller value of distance and the results will be send to the first layer. The PEs in the first layer also find the smaller value and send the results to the centralized processing element. The zero level determine the smallest of its input pairs which is shown in Figure 4.

3. Analysis and Time Complexity

The diagrammatic representation in Figure 5 shows the phases of the procedure for the proposed ranking algorithm.

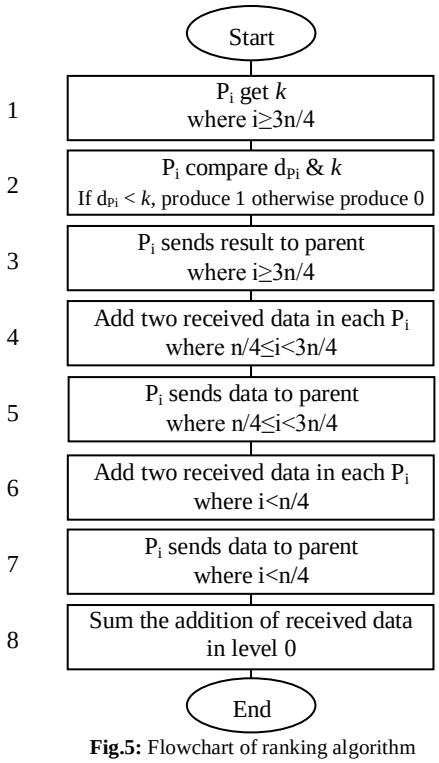

Fig.5: Flowchart of ranking algorithm

The execution time of ranking and closest element is constant. In the ranking algorithm, in the second step of the flowchart just one comparison is performed to produce one or zero, which its time execution is *O(1)*. In the forth step in the flowchart, each PE perform addition operation while receive its data. This operation occurs in *O(1)*. The operations in the sixth step is the same as the forth step which requires *O(1)* execution time. The last step needs three additions to calculate the rank of the chosen value in the centralized node which is performed in *O(3)*.

The time for execution of the algorithm is constant which is $t_n=O(6)=O(1)$, hence the cost of the algorithm is the number of PEs $c_n=O(N)$.
The execution time of closest element algorithm is like the ranking algorithm. The operations are just different. The operation in the third layer is subtraction in this algorithm which is $O(1)$. The second and first layer PEs perform a comparison with $O(1)$ execution time. The zero level carry out three comparison which needs $O(3)$. The total time complexity is $t_n=O(6)=O(1)$, as a result the total cost will be $c_n=O(N)$ which is the number of PEs.

4. Conclusion

There are different types of search algorithms which are known for different purposes such as position, rank, count and closest element. The position and count algorithms were described in our previous work on our architecture named Centralized Diamond. In this paper, we proposed ranking algorithm and closest element algorithm on the centralized diamond architecture with constant execution time which have appropriate time complexities.

References

[1] D. Eppstein, Z. Galil, "Parallel algorithmic techniques for combinational computation", *Annual review of computer science*, Vol. 3, No. 1, (1988), pp. 233-283.

[2] R. Sarnath, X. He, "Efficient parallel algorithms for selection and searching on sorted matrices", *Proceedings of Sixth International Parallel Processing Symposium*, (1992), pp. 108-111.

[3] K. A. Berman, J. L. Paul, *Algorithms: sequential, parallel, and distributed*, Course Technology Ptr, (2005).

[4] K.T. Yar, N.S.S Aung, "The optimal running time from computational models by comparing two parallel searching procedures", *IEEE International Workshop on Open-source Software for Scientific Computation (OSSC)*, (2009), pp. 73-78.

[5] M. Damrudi, K. Jadidy Aval. "A Parallel Search on Hypercube Interconnection Network", *Journal of Computer Science & Computational Mathematics*, Vol.2, No. 1, (2012).

[6] M. Damrudi, K. Jadidy Aval, "Searching data using centralize diamond architecture", *Journal of Communication and Computer*, Vol. 8, No. 9, (2011), pp. 807-811.

[7] Masumeh, D. and A. Kamal Jadidy, Utilizing Centralized Diamond Architecture for Searching Algorithms. Journal of Physics: Conference Series, 2017. 892(1): p. 012001.

[8] Akl, S.G., *The design and analysis of parallel algorithms*, Prentice-Hall, Inc. 401, (1989).

[9] M. Damrudi, K.J. Aval, Diamond architecture with NOD sorting, *IEEE Information Computing and Telecommunications*, (2009).

[10] M. Damrudi, K.J. Aval, A new parallel sorting on diamond architecture, *Applied Electromagnetics, Wireless & Optical Communications (Electroscience '10)*, (2010)